**NAME (as it appears on your UF ID): _____**

(Please **PRINT**)

**UF Student ID#: _____**

---------- CEN 6070 Software Testing & Verification ----------

Exam 2 – Summer 2016

You have 90 minutes to work on this exam. It is a "closed-book/closed-notes" test. Pay attention to point values, since you may not have time to complete all 11 problems. **You should assume that all variables represent INTEGERS, unless otherwise indicated.**

PRINT your name above NOW and sign the pledge at the bottom of the last page, if appropriate, when you are finished.

You will be given "scratch paper," but PLEASE PRINT ANSWERS IN THE SPACE PROVIDED ON THE EXAM **(EXCLUDING MARGINS)** ONLY– PREFERABLY USING A BALLPOINT PEN TO INCREASE LEGIBILITY. Good luck!

1. (16 pts.) Consider the assertion of ***weak*** correctness: $\{a=b \vee b>0\}\ S\ \{a+b=c \wedge b<0\}$. Which of the following observations/facts **would** allow one to deduce that the assertion is false, and which **would not**? Circle either "would" or "would not" as appropriate, considering the observations individually. To compensate for random guessing, your score in points will be 2 times the number of [correct minus incorrect] answers, or 0 – whichever is greater. Therefore, if you are not more than 50% sure of your answer, consider skipping the problem.

   a. sp(S, b=5) = (a=c) — would / would not

   b. If a=-5 and $b^2$=25 prior to the execution of S, then S terminates with c=17. — would / would not

   c. wlp(S, a>c) = (ab<0 ∧ a>0) — would / would not

   d. wp(S, a<c) = (ab>0 ∧ a>0) — would / would not

   e. When the initial value of b is 17, S does not terminate. — would / would not

   f. S *sometimes* terminates with b=17 when the initial value of b is -17. — would / would not

   g. S *always* terminates with b=17 when the initial value of b is -17. — would / would not

   h. wlp(S, a+b=c ∧ b<0) ≠ (a=b ∨ b>0) — would / would not

2. (18 pts.) Circle either "true" or "false" for each of the following assertions. To compensate for random guessing, your score in points will be 2 times the number of [correct minus incorrect] answers, or 0 – whichever is greater. Therefore, if you are not more than 50% sure of your answer, consider skipping the problem.

a. If K = wp(S, Z), then sp(S,K) $\Rightarrow$ Z. true false

b. If K $\Rightarrow$ wp(S, Z), then sp(S,K) = Z. true false

c. Assuming S terminates whenever K holds initially, if sp(S,K) $\Rightarrow$ Z, then K = wp(S,Z). true false

d. Assuming S terminates whenever K holds initially, if sp(S,K) = Z, then K $\Rightarrow$ wp(S,Z). true false

e. Suppose K = wp(while b do S, Q). Then K is a Q-adequate loop invariant for {K} while b do S {Q}. true false

f. {P} while b do S {Q} $\Leftrightarrow$

```
{P}
    if b then
        repeat S until NOT b
    end_if
{Q}
```

true false

g. The Method of Well Founded Sets, as presented in class, *cannot* be used to prove that the program below will terminate for some initial values of x,y. true false

```
while (y>0) do
   y := y+x
   if (x≥0) then
        x := x-1
   end_if
end_while
```

h. Formally speaking, Z=XJ is a **loop invariant** for the assertion: true false

```
{Z=0 ∧ X=0}
    while J<Y do
        Z := Z+X;
        J := J+1
    end_while
{Z=XY}
```

i. Z=XJ ∧ J≤Y is a **Q-adequate loop invariant** for the assertion given in part (h) above. true false

3. a. (2 pts.) Complete the ROI for proving the weak correctness of program S with respect to pre-condition P and post-condition Q using the weakest liberal pre-condition (wlp) predicate transform:

$$\frac{\qquad\qquad}{\{P\}\ S\ \{Q\}}$$

b. (4 pts.) Identify expressions for $H_1$, $H_2$, $H_3$, and $H_k$ such that

wp(Repeat s Until c, Q) = $H_1$ V $H_2$ V $H_3$ V...V $H_k$ V...

where $H_i$ represents the necessary and sufficient condition that "Repeat s Until c" terminates in state Q after i executions of s. (Hint: for i>1, $H_i$ should be expressed as a function of $H_{i-1}$.)

$H_1$:

$H_2$:

$H_3$:

$H_k$:

c. (15 pts.) Find the weakest liberal pre-condition (wlp) of the program:

K: Repeat z := z+y; t := t-1 Until t=0

with respect to the post-condition {z=yx}. (Give the values of $H_1$, $H_2$, and $H_k$, where the **wp**(K, z=yx) is given by the infinite expression: $H_1$ V $H_2$ V ... V $H_k$ V ... and then express the **wlp**(K, z=yx) in closed form.) ASSUME that wp(K, true) is t>0.

$H_1$:

$H_2$:

$H_3$:

$H_k$:

Closed form expression of **wlp**: _____

3. (cont'd)

d. (6 pts.) Use the ROI from part (a) and the closed form expression of the wlp from part (c) to prove the assertion:

{t=2 Λ x=6 Λ z=8 Λ y=2} Repeat z := z+y; t := t-1 Until t=0 {z=yx}

Note that, by observation, the values of both x and y are invariant with respect to program K.

4. (8 pts.) Given *P1, P2, f1,* and *f2:*

*P1* = Repeat z := z∗x; x := x-1 Until x=1

*P2* = Repeat x := x-1; z := z∗x until x<3

*f1* = (x>2 -> x, z := 1, zx!)

*f2* = (x>2 -> x, z := 2, z(x-1)! | x=1 -> x, z := 0, 0 | true -> x, z := x-1, z(x-1))

Determine the correctness relationships among the given programs and functions. In the table below, indicate "C" for Complete program correctness, "S" for Sufficient program correctness only, and "N" for Neither. To compensate for random guessing, you will receive +2 pts. for each correct answer given and -1 pt. for each incorrect answer given, with a minimum possible score of 0 pts.

| | *P1* | *P2* |
|---|---|---|
| *f1* | | |
| *f2* | | |

5. (20 pts.) Circle either "valid" or "invalid" for each of the following *hypothesized* Rules of Inference. To compensate for random guessing, your score in points will be 2 times the number of [correct minus incorrect] answers, or 0 – whichever is greater. Therefore, if you are not more than 50% sure of your answer, consider skipping the problem.

a. $\dfrac{\{true\}\ \text{if } b \text{ then } S\ \{\neg b \wedge Q\}\ \text{strongly}}{\{P\}\ \text{while } b \text{ do } S\ \{Q\}}$ ? valid invalid

b. $\dfrac{\{true\}\ \text{if } b \text{ then } S\ \{b \vee Q\}}{\{P\}\ \text{while } b \text{ do } S\ \{Q\}\ \text{strongly}}$ ? valid invalid

c. $\dfrac{\{true\}\ S\ \{K\}\ \text{strongly},\ \{K\}\ \text{while } \neg b \text{ do } S\ \{Q\}}{\{P\}\ \text{repeat } S \text{ until } b\ \{Q\}\ \text{strongly}}$ ? valid invalid

d. $\dfrac{sp(S, P) \Rightarrow Q}{\{P\}\ S\ \{Q\}\ \text{strongly}}$ ? valid invalid

e. $\dfrac{I \Rightarrow P,\ \{I\}\ S\ \{I\},\ P \Rightarrow Q}{\{P\}\ \text{repeat } S \text{ until } b\ \{Q\}}$ ? valid invalid

f. $\dfrac{(\neg b) \Rightarrow Q}{\{true\}\ \text{while } b \text{ do } S\ \{Q\}}$ ? valid invalid

g. $\dfrac{\{true\}\ S\ \{I\}\ \text{strongly},\ (I \wedge b) \Rightarrow Q}{\{P\}\ \text{repeat } S \text{ until } b\ \{Q\}\ \text{strongly}}$ ? valid invalid

h. $\dfrac{Q \Leftrightarrow sp(S, P)}{\{true\}\ S\ \{Q\}}$ ? valid invalid

i. $\dfrac{(wlp(S, Q) \wedge K) \Leftrightarrow P}{\{P\}\ S\ \{Q\}}$ ? valid invalid

j. $\dfrac{\{P \wedge b\}\ S\ \{I\},\ ((I \vee P) \wedge \neg b) \Rightarrow Q}{\{P\}\ \text{while } b \text{ do } S\ \{Q\}}$ ? valid invalid

6. (11 pts.) Prove the assertion of weak correctness below using the appropriate loop Rule of Inference with the invariant: z=y(x-t). SHOW AND JUSTIFY ALL STEPS AND CASES AS ILLUSTRATED IN CLASS.

```
{true}
   z := 0
   t := x
   Repeat
      z := z+y
      t := t-1
   Until t=0
{z=yx}
```

7. (23 pts.) For program *H* and intended program function *f* given below, Prove *f* = [*H*] by showing that the Repeat_Until complete correctness conditions hold. You may assume (i.e., you need NOT prove) that the function of the loop body, *g*, is (t,z := t-1,z+y). STATE AND **PROVE** ALL OTHER CONDITIONS, STEPS, AND CASES AS ILLUSTRATED IN CLASS.

*H:* Repeat
z := z+y
t := t-1
Until t=0

*f* = (t>0 → t,z := 0,z+yt)

(Continue your proof on the next page if necessary.)

7. (cont'd)

8. In Lecture Notes #18 we considered heuristics for synthesizing Q-adequate loop invariants that could be used with the while loop ROI to prove assertions of the form {P} while b do s {Q}. Later in the course we learned that a loop computing a function, *f,* maintains a very important property of state across iterations.

   a. (5 pts.) Describe this property algebraically and explain what it means.

   b. (5 pts.) In Lecture Notes #24 we saw examples of how this property may yield a Q-adequate loop invariant that can be used prove {P} while b do s {Q}. Under what specific circumstances would this approach realistically eliminate the need for heuristics to synthesize Q-adequate loop invariants to prove such assertions?

9. (10 pts.) Assume that for some unknown initial ("input") states $X_a = (x_a, y_a)$ and $X_b = (x_b, y_b)$, two while loops computing functions *t* and *s* terminate with final ("output") states (0, 7) and (-2, -9), respectively; i.e., $t(X_a) = (0,7)$ and $s(X_b) = (-2,-9)$. Also, assume that

$$q_t(X) \text{ is } y=y_0+2(x_0-x) \quad \text{and} \quad q_s(X) \text{ is } y=y_0+3(x-x_0).$$

   Can it be determined whether or not there is some **common** initial or intermediate state generated by the two loops in computing these "outputs" based on the information given above? If so, and if such a state exists, identify it. If not, briefly describe what additional information would be needed to do so.

10. (10 pts.) The following statements relate to King, et al., "Is Proof More Cost Effective than Testing?" Indicate whether each is true or false. To compensate for random guessing, your score in points will be 2 times the number of [correct minus incorrect] answers, or 0 – whichever is greater. Therefore, if you are not more than 50% sure of your answer, consider skipping the problem.

    a. The paper describes the use of formal development methods on an industrial safety-critical application. true false

    b. The "SPARK" notation was used for documenting the system specification and part of the design, and a subset of Ada was used for coding. true false

    c. Z-proofs appeared to be substantially more efficient at finding faults than the most efficient testing phase. true false

    d. The authors found that a good (informal) understanding of the meaning of imperative programming constructs, together with some familiarity with relevant proof concepts, such as loop invariants, were necessary to conduct the formal code proofs. true false

    e. Traditional Unit and Integration Testing did find a number of faults involving some critical numerical calculations. true false

11. (10 pts.) In "Cleanroom Software Engineering for Zero-Defect Software," Linger argues that ***statistical usage testing*** is many times more effective at extending ***MTTF*** than is ***coverage testing***. Briefly describe what these terms mean and why, according to Linger, this is the case.

On my honor, I have neither given nor received unauthorized aid on this exam and I pledge not to divulge information regarding its contents to those who have not yet taken it. _____

SIGNATURE